Lehrstuhl Informatik 4
Andreas Blume, Egor Kudrjaschow
Nils Dunker, Bastian Korte, Minsu So

technische universität dortmund

Wintersemester 2020/2021

Rechnernetze und verteilte Systeme
Programmieraufgabe 3: Switch

Ausgabe: Sa. 19.12.2020; Abgabe: Di. 19.1.2021

Revision 1, 29.12.2020: Korrekturen in den Abschnitten 1.1 und 4.2 sind rot unterlegt.

1 Allgemeine Hinweise zur Bearbeitung

Die Programmieraufgaben sollen in Gruppen mit **maximal fünf Studierenden** bearbeitet werden. Die Implementierung soll in **Java 11** geschehen. Machen Sie sich also zunächst mit der Programmierung in Java (siehe z. B. offizielle Dokumentation der Java API) vertraut.

1.1 Bibliotheken

Zur Lösung der Programmieraufgaben soll ausschließlich auf die durch die Java-Standardbibliothek zur Verfügung gestellten Klassen zurückgegriffen werden, die sich in den folgenden Paketen (oder Subpaketen davon) befinden:

- `java.io.*`
- `java.lang.*`
- `java.net.*`
- `java.nio.*`
- `java.security.*`
- `java.util.*`
- `java.text.*`
- `java.time.*`

Klassen aus den Pakten `com.sun.net.*` bzw. `sun.net.*` oder Bibliotheken von Drittanbietern dürfen nicht eingebunden werden. Ebenfalls ist das Kopieren von Quellcode aus Bibliotheken oder anderen Fremdquellen nicht erlaubt. **Verstöße führen zum Nichtbestehen der Studienleistung!**

1.2 Fehlerbehandlung

Im Programm sollten Exceptions (bzw. Fehlerzustände), die beispielsweise bei arithmetischen Operationen (Division durch 0) oder bei Zugriff auf noch nicht erzeugte Objekte entstehen können, verarbeitet werden. Eine unzureichende Fehlerbehandlung führt zu Punktabzug.

1.3 Kommentare und Dokumentation

Kommentieren Sie Ihren Quelltext ordentlich! Für Abgaben ohne ausreichende Kommentare werden Punkte abgezogen. Zusätzlich sollten Sie alle über die Aufgabenstellung hinausgehenden Bedienungsaspekte der Applikation (Befehle, Kommandozeilenparameter, ...) dokumentieren, um eine reibungslose Korrektur zu ermöglichen. Neben einer Textdatei ist es auch zulässig, diese Dokumentation von der Anwendung selbst (z. B. direkt nach dem Starten) ausgeben zu lassen.

1.4 Abgabe

Die Abgabe findet im Moodle statt. Die Abgabe kann bis zum Abgabetermin beliebig oft geändert werden, wobei die zuletzt abgegebene Version bewertet wird.

Die Abgabe muss vom Compiler übersetzbar sein und unter **Java 11** laufen. Abzugeben ist ein zip-Archiv mit allen zum Projekt gehörenden Quelltexten und eine ausführbare jar-Datei.

2 Tipps zur Programmierumgebung

2.1 IDE

Ein hilfreiches Werkzeug bei der Entwicklung von Software ist eine *integrierte Entwicklungsumgebung* (IDE). Bekannte Java-IDEs sind:

- **IntelliJ IDEA**
- **Eclipse**
- NetBeans IDE
- Editoren mit Java Plugin wie
  - Visual Studio Code
  - Atom
  - viele weitere

Alle genannten IDEs verfügen über detaillierte Anleitungen zur Installation und dem Import von Projekten.

2.2 Git Repositories

Git ist eine freie Software zur verteilten Versionsverwaltung von Dateien, die die gemeinsame Arbeit an den Programmieraufgaben unterstützen kann. Eine kurze Anleitung zu Git finden Sie z. B. hier. Einige Remote Git Server sind:

- (Fakultät/IRB) Gitea

- (Fachschaft) GitLab
- (Microsoft) GitHub
- (Atlassian) Bitbucket

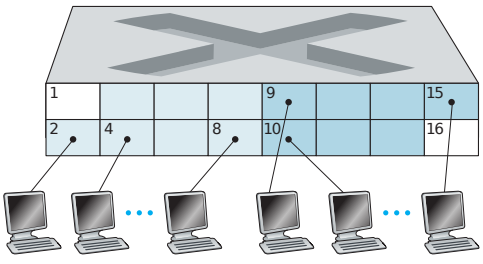
Wichtig: Erstellen Sie *private* Repositories um Plagiate zu vermeiden! Öffentliche Repositories werden von Internet-Suchmaschinen (z. B. Google) indexiert und können somit von anderen Gruppen gefunden werden.

3 Aufgabe

Switches sind Kopplungsgeräte in Rechnernetzen, die Netzwerksegmente miteinander verbinden und das ISO/OSI-Basisreferenzmodell bis zur Schicht 2, Sicherungsschicht, implementieren.[1] Ihre Aufgabe besteht aus dem Filtern und Weiterleiten von an den Ports eingehenden Frames. Das Weiterleiten und Filtern erfolgt mithilfe einer Switch-Tabelle (engl. Source-Address-Table, SAT). Diese Tabelle lernt der Switch automatisch, dynamisch und autonom (siehe [KR17, Abschnitt 6.4]). Switches sind also selbstlernend.

Eigenständiges Lernen der Switch-Tabelle:

1) Die Switch-Tabelle ist zunächst leer.
2) Bei jedem auf einem Port eingehenden Frame, speichert der Switch:
   - die MAC-Adresse aus dem Feld Quelladresse des Frames
   - den Port, über den der Frame eintraf
   - die aktuelle Zeit
3) Der Switch löscht eine MAC-Adresse aus der Tabelle, wenn nach einem bestimmten Zeitraum keine Frames empfangen wurden, die diese MAC-Adresse als Quelladresse enthielten.



Ein Switch mit 16 Ports. [KR17]

MAC-Adresse	Port	Zeit
0B:36:B0:B9:E0:81	1	12:00:00
AF:D8:DC:73:A9:0F	3	12:02:00
E5:8C:25:6F:A5:2D	10	12:05:00
8A:9F:D8:D2:91:72	5	12:18:00
...	...	...

Tabelle 1: Ausschnitt einer Switch-Tabelle.

[1] Eine Ausnahme bilden „Layer 3 Routing Switches“. Sie implementieren das ISO/OSI-Basisreferenzmodell bis zur Vermittlungsschicht.

Nehmen wir nun an, dass am Port p des Switches ein Frame ankommt, der die Zieladresse MAC_{ziel} besitzt. Dann können die folgenden drei Szenarien unterschieden werden.

- Es gibt keinen Eintrag für MAC_{ziel} in der Switch-Tabelle. In diesem Fall erfolgt eine Ausgabe des Frames auf allen Ports außer Port p.
- In der Switch-Tabelle existiert ein Eintrag für MAC_{ziel} mit Port p. Der Eingangsport und der Ausgangsport sind also identisch. Demzufolge findet eine Filterung des Frames statt und der Frame wird vom Switch verworfen.
- Die Switch-Tabelle enthält einen Eintrag für MAC_{ziel} mit Port $x \neq p$. Folglich kommt es zu einer Weiterleitung, bei der der Frame auf Port x ausgegeben wird.

Dazu zwei kurze Beispiele. Am Switch-Port 10 kommt ein Frame mit der Zieladresse E5:8C:25:6F:A5:2D an. Der Switch müsste also laut Tabelle 1 den Frame auf Port 10 weiterleiten. Da dies jedoch dem Eingangsport des Frames entspricht, wird der Frame vom Switch gefiltert und verworfen. Wäre derselbe Frame auf Port 20 des Switches angekommen, dann käme es zu einer Weiterleitung auf den Switch-Port 10.

3.1 Switching-Engine

Die vorgestellte Switch-Funktionalität soll in dieser Programmieraufgabe in einem vereinfachten Rahmen realisiert werden. Das zu entwickelnde Java-Programm ist also eine Switching-Engine.

Wir stellen uns einen Switch mit N Ports vor und nehmen zur Vereinfachung maximal 255 Adressen an, wobei die Adresse 255 als Broadcast-Adresse verwendet wird. Der Eingang eines Frames am Switch erfolgt durch eine Eingabe im folgenden Format.

`frame <Eingangsportnummer> <Absenderadresse> <Zieladresse>`

Die Eingangsportnummer, die Absenderadresse und die Zieladresse sind Integer und haben die folgenden Definitionsbereiche:

- Eingangsportnummer: 1 bis N
- Absenderadresse: 1 bis 254
- Zieladresse: 1 bis 255

Die Switching-Engine liest anschließend die Eingabe und entscheidet entsprechend des Zustands der Engine, welche Folgeaktion für den Frame notwendig ist und gibt diese aus. Außerdem kommt es gegebenfalls zu einer Aktualisierung der Switch-Tabelle.

3.2 Weitere Kommandos

Neben der Verarbeitung von eingehenden Frames am Switch soll das zu entwickelnde Java-Programm die Möglichkeit bieten die aktuellen Einträge der Switch-Tabelle und die Portnutzung anzuzeigen. Darüber hinaus soll das Löschen von veralteten Tabellen-Einträge simuliert werden.

Die zu implementierenden Kommandos sind:

- `table`
  → Gibt die aktuellen Einträge der Switch-Tabelle aus.
- `statistics`
  → Gibt an, wie oft die einzelnen Ports bisher verwendet wurden.
- `del Xs` bzw. `del Xmin`
  → Löscht alle Switch-Tabelleneinträge, die älter als X Sekunden bzw. X Minuten sind, wobei $X > 0$ eine ganze Zahl ist. Beispiele: `del 80s` und `del 2min`
- `exit`
  → Programm beenden.

3.3 Fehlerhafte und ungültige Eingaben

Alle Nutzereingaben, die mit keinem zuvor beschrieben Format übereinstimmen, sollen vom Java-Programm mit der Fehlermeldung „*Ungültige Eingabe!*“ beantwortet werden.

4 Beispiele und Command Line Interface

Es folgen einige Beispiele, die sowohl die Funktionsweise der Switching-Engine als auch das Command Line Interface zeigen.

Das Java-Programm soll zunächst vom Nutzer die Anzahl der Ports N entgegennehmen, wobei die Nutzereingabe nur ein Integer größergleich 1 sein darf. Ungültige Eingaben führen zu einer passenden Fehlermeldung und das Programm wird beendet. Anschließend werden, falls notwendig, die benötigten Datenstrukturen angelegt und es erfolgt die Ausgabe „*Ein N-Port-Switch wurde erzeugt.*“. Nun kann der Nutzer die Switching-Engine verwenden. Dabei gibt das Programm jeweils “`$ `“ aus, anschließend folgt die Eingabe des Nutzers. Diese Eingaben müssen zunächst auf Korrektheit geprüft werden. Falsche Eingabe führen zu einer Fehlermeldung, Korrekte werden verarbeitet und beantwortet.

Die jeweiligen Antworten der Switching-Engine sind in den folgenden Beispielen angegeben. Das Antwortformat für `table` und `statistics` sind Tabellen, wobei die `table`-Antwort nach der Adressse und die `statistics`-Tabelle nach Port aufsteigend sortiert sind. Wenn die Switch-Tabelle (noch) leer ist, gibt das Programm „*Die Switch-Tabelle ist leer.*“ aus. Die `statistics`-Tabelle enthält immer alle Ports. Die Portnutzung wird zu Beginn mit 0 initialisiert. Es werden pro Port alle eingehenden und ausgehenden Frames gezählt.

4.1 2-Port-Switch

Ein 2-Port-Switch besteht aus zwei Ports. Der Nutzer gibt also hier für die Anzahl Ports 2 ein. Der 2-Port-Switch wird erzeugt und kann dann vom Nutzer verwendet werden.

```
Anzahl Ports: 2   // "Anzahl Ports: " ist eine Programmausgabe, "2" eine Eingabe

Ein 2-Port-Switch wurde erzeugt.

$ frame 1 22 33   // "$ " ist eine Programmausgabe, "frame 1 22 33" eine Eingabe
Ausgabe auf allen Ports außer Port 1. // Grund: 33 nicht in der Switch-Tabelle

$ table           // "$ " ist eine Programmausgabe, "table" eine Eingabe
Adresse Port Zeit
     22    1 12:00:00 // Zeitformat: HH:mm:ss

$ statistics      // "$ " ist eine Programmausgabe, "statistics" eine Eingabe
Port Frames       // Tabelle nach Port aufsteigend sortiert
   1      1       // alle Ports ausgeben
   2      1       // bei bisher nicht genutzten Ports gilt Frames = 0

$ frame 1 60 22   // "$ " ist eine Programmausgabe, "frame 1 60 22" eine Eingabe
Frame wird gefiltert und verworfen. // Grund: Eingangsport = Ausgangsport = 1

$ table
Adresse Port Zeit // Tabelle nach Adresse aufsteigend sortiert
     22    1 12:00:00
     60    1 12:01:00

$ statistics
Port Frames
   1      2
   2      1

$ frame 2 60 22     // Switch-Tabelle aktualisieren: Adresse 60 jetzt am Port 2
Ausgabe auf Port 1. // Grund: Adresse 22 am Port 1, Port 2 ungleich Port 1

$ table
Adresse Port Zeit
     22    1 12:00:00
     60    2 12:02:00

$ statistics
Port Frames
   1      3
   2      2

$ frame 3 -1 500
Ungültige Eingabe! // Gründe: Port > 2, Absenderadresse < 1, Zieladresse > 255

$ exit
Das Programm wird beendet.
```

4.2 6-Port-Switch

Dieses Beispiel zeigt die Verwendung der Broadcast-Adresse und das Löschen von veralteten Einträgen. Das Löschen von Einträgen in der Switch-Tabelle hat keine Auswirkung auf die `statistics`-Tabelle. Wenn das Kommando `del` zu keiner Änderung in Switch-Tabelle führt, dann soll vom Java-Programm „*Es wurden keine Adressen aus der Switch-Tabelle gelöscht.*“ ausgegeben werden.

```
Anzahl Ports: 6  // "Anzahl Ports: " ist eine Programmausgabe, "6" eine Eingabe

Ein 6-Port-Switch wurde erzeugt.

$ abc231
Ungültige Eingabe!

$ frame 1 23 54
Ausgabe auf allen Ports außer Port 1.

$ frame 4 32 23
Ausgabe auf Port 1.

$ frame 6 35 32
Ausgabe auf Port 4.

$ frame 2 32 23
Ausgabe auf Port 1.

$ frame 6 85 32
Ausgabe auf Port 2.

$ frame 1 55 35
Ausgabe auf Port 6.

$ frame 6 5 255   // Achtung: 255 ist die Broadcast-Adresse
Broadcast: Ausgabe auf allen Ports außer Port 6.

$ table
Adresse Port Zeit
      5    6 10:05:00
     23    1 10:00:00
     32    2 10:03:00
     35    6 10:02:00
     55    1 10:04:30
     85    6 10:04:00

$ del 3min   // um 10:05:30 ausgeführt: alle Einträge älter als 10:02:30 löschen
Folgende Adressen wurden aus der Switch-Tabelle gelöscht: 23, 35
```

```
$ table
Adresse Port Zeit
      5    6 10:05:00
     32    2 10:03:00
     55    1 10:04:30
     85    6 10:04:00

$ statistics
Port Frames
   1      5
   2      4
   3      2
   4      4
   5      2
   6      5

$ exit
Das Programm wird beendet.
```

Literatur

[KR17] J. Kurose and K. Ross. *Computer Networking: A Top-Down Approach, Global Edition*. Pearson Education Limited, 2017.